



# POLITECNICO MILANO 1863

Computer Science and Engineering

## Code Inspection

February 2, 2017

Prof. Luca Mottola

Authors:

- ZHOU YINAN(Mat. 872686)
- ZHAO KAIXIN(Mat. 875464)
- ZHAN YUAN(Mat. 806508)

# Contents

|          |   |          |
|----------|---|----------|
| <b>1</b> | <b>Classes Assigned</b>                             | <b>3</b> |
| <b>2</b> | <b>Functional Role</b>                              | <b>4</b> |
| 2.1      | ProductionRun.java . . . . .                        | 4        |
| 2.2      | ViewerServletRequest . . . . .                      | 5        |
| <b>3</b> | <b>Check List</b>                                   | <b>8</b> |
| 3.1      | ProductionRun.java . . . . .                        | 8        |
| 3.1.1    | Naming Conventions . . . . .                        | 8        |
| 3.1.2    | Indention . . . . .                                 | 8        |
| 3.1.3    | Braces . . . . .                                    | 8        |
| 3.1.4    | File Organization . . . . .                         | 8        |
| 3.1.5    | Wrapping Lines . . . . .                            | 8        |
| 3.1.6    | Comments . . . . .                                  | 8        |
| 3.1.7    | Java Source Files . . . . .                         | 8        |
| 3.1.8    | Package and Import Statements . . . . .             | 8        |
| 3.1.9    | Class and Interface Declarations . . . . .          | 8        |
| 3.1.10   | Initialization and Declarations . . . . .           | 8        |
| 3.1.11   | Method Calls . . . . .                              | 8        |
| 3.1.12   | Arrays . . . . .                                    | 8        |
| 3.1.13   | Object Comparison . . . . .                         | 8        |
| 3.1.14   | Output Format . . . . .                             | 8        |
| 3.1.15   | Computation, Comparisons, and Assignments . . . . . | 8        |
| 3.1.16   | Exceptions . . . . .                                | 8        |
| 3.1.17   | Flow of Control . . . . .                           | 8        |
| 3.1.18   | Files . . . . .                                     | 8        |
| 3.2      | ViewerServletRequest . . . . .                      | 9        |
| 3.2.1    | Naming Conventions . . . . .                        | 9        |
| 3.2.2    | Indention . . . . .                                 | 9        |
| 3.2.3    | Braces . . . . .                                    | 9        |
| 3.2.4    | File Organization . . . . .                         | 9        |
| 3.2.5    | Wrapping Lines . . . . .                            | 9        |
| 3.2.6    | Comments . . . . .                                  | 9        |
| 3.2.7    | Java Source Files . . . . .                         | 9        |
| 3.2.8    | Package and Import Statements . . . . .             | 9        |
| 3.2.9    | Class and Interface Declarations . . . . .          | 9        |
| 3.2.10   | Initialization and Declarations . . . . .           | 9        |
| 3.2.11   | Method Calls . . . . .                              | 9        |
| 3.2.12   | Arrays . . . . .                                    | 9        |

|        |   |    |
|--------|---|----|
| 3.2.13 | Object Comparison . . . . .                     | 9  |
| 3.2.14 | Output Format . . . . .                         | 9  |
| 3.2.15 | Computation, Comparisons, and Assignments . . . | 9  |
| 3.2.16 | Exceptions . . . . .                            | 9  |
| 3.2.17 | Flow of Control . . . . .                       | 10 |
| 3.2.18 | Files . . . . .                                 | 10 |

## 1 Classes Assigned

We have been assigned two classes :

- ProductionRun.java
- ViewerServletRequest.java

The namespace patten is :

```
../apache-ofbiz-16.11.01/applications/manufacturing/src/main/java/org/apache/ofbiz  
/manufacturing/jobshopmgt/ProductionRun.java
```

```
../apache-ofbiz-16.11.01/specialpurpose/birt/src/main/java/org/apache/ofbiz/birt  
/report/servlet/ViewerServletRequest.java
```

## 2 Functional Role

### 2.1 ProductionRun.java

Instead of directly looking into the code, we first examine the online ofbiz document to get information of this class. This class belongs to **Manufacturing** section. The link to the Online Documet.

The situation is described here. After a client makes order, configurable goods which our company provide require some type of manufacturing or production. If we do not have the requiring parts in our inventory, a production run is generated.

The screenshot shows a web browser window displaying the 'OFBiz Project Open Wiki' page. The page title is 'Beginner's Guide to the Manufacturing Process'. It includes a sidebar with a navigation menu and a main content area with text and a table.

Access to add and change pages is restricted. See: <https://wiki.apache.org/confluence/display/OFBIZ/No+Spam>

### OFBiz Project Open Wiki / ... / Manufacturing

## Beginner's Guide to the Manufacturing Process

Created by Unknown User (qhorton), last modified on Mar 24, 2009

**"This is still a WIP. More to come"**

#### Beginner's Guide to the Manufacturing Process

This page will cover the basic aspects of the Manufacturing Process from a semi-technical point of view. It will use the configurable PC(PC001) which is part of the demo seed data. Hopefully, this will be a supplement to the other pages on this topic and will find a nice home amongst them.

To begin, login to the ecommerce application and configure the PC001 with all of the extra options. These include the 2 Gb RAM, 500 Gb HDD, second 500 Gb HDD, NIC, and modem. Do a checkout to complete the order.

Before we continue a little analogy....

When a customer orders a configurable good, some degree of manufacturing must occur to produce the finished product. Imagine that we just ordered this computer from Dell. Dell has to gather all of the various components for the computer, build the computer, test the computer, pack it into a box, and ship it. Add a half dozen or so in-between steps(e.g. transferring parts from a warehouse or moving the computer from the building lab to the shipping room) in there and you have the whole manufacturing process. Think of this scenario when you are reading through this page so you have a mental image to connect things to.

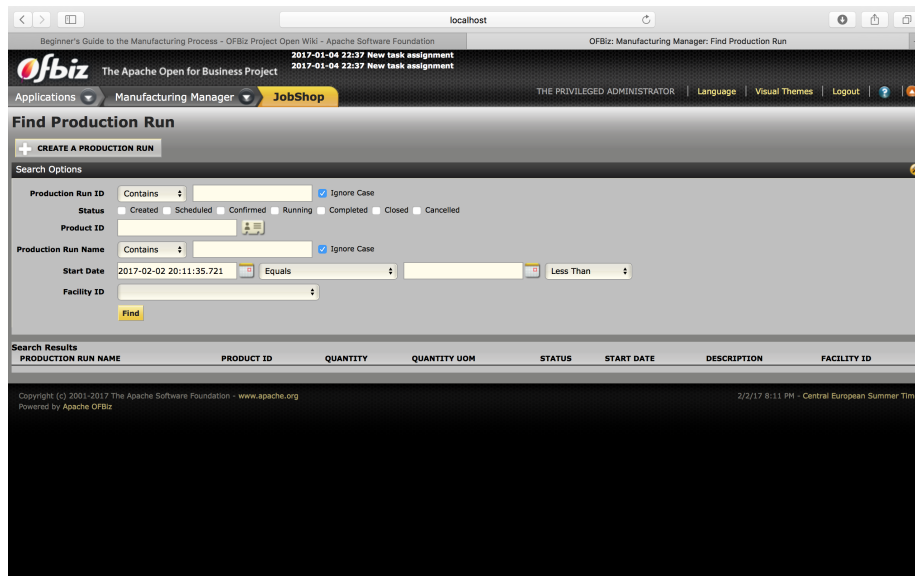
Now that you have placed the order, log into the backend of OFBiz: <http://localhost:8080/webtools>. If you click on the order tab and then your newly created order, it will pull up the detailed sales order form. Scroll down to the Order Items Screenlet. The first thing you will notice is that a new Product ID of 10010(May be different in your install) has been created for the Configurable PC. We will come back to this shortly. Below this are the following fields:

| Product                          | Status           | Quantity | Unit / Unit | Adjustments         | Sub Total |
|----------------------------------|------------------|----------|-------------|---------------------|-----------|
| 20010 - Configurable PC          | Current Approved | Ordered  | 1           | \$1,153.00 / \$5.00 | \$5.00    |
| 2009-03-24 01:09:51.167 Approved |                  |          |             |                     |           |

Powered by a free **Atlassian Confluence Open Source Project License** granted to Apache Software Foundation. Evaluate Confluence today.  
This Confluence installation runs a Free Gifty License - Evaluate the Gifty Confluence Plugin for your Wiki!

Powered by Atlassian Confluence 5.8.4, Team Collaboration Software · Report a bug · Atlassian News

If we log into the ofbiz web application, we can examine the production run section. The ProductionRun class manages all the information of a certain production run activity.

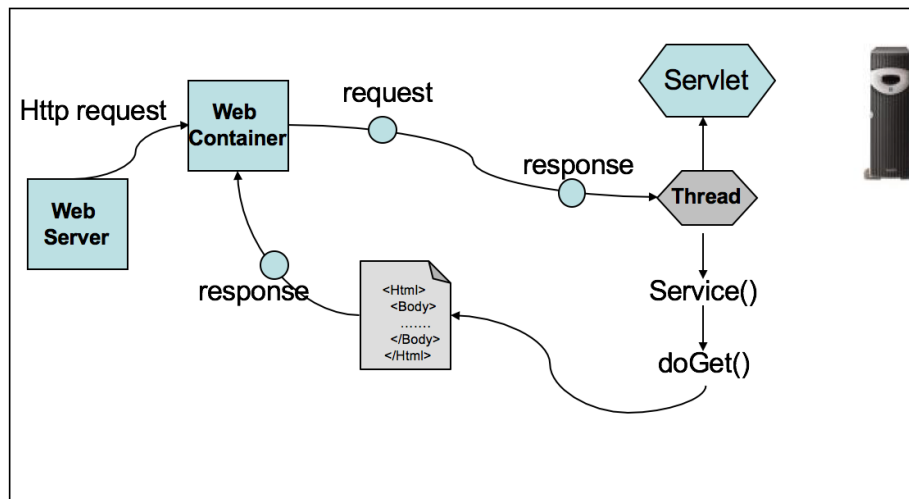
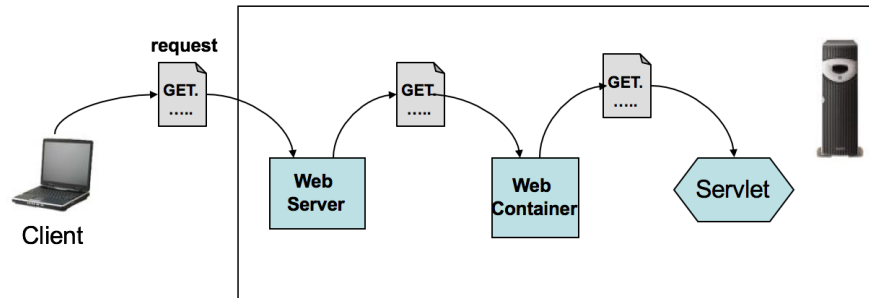


## 2.2 ViewerservletRequest

By looking at the code, we find that `ViewerservletRequest` extends `HttpServletRequestWrapper`. A "`HttpServletRequestWrapper`" provides a convenient implementation of the `HttpServletRequest` interface that can be subclassed by developers wishing to adapt the request to a Servlet. Thus the role of this class is to represent a specific function of `HttpServletRequest`. More specifically, this function is `getParameter()`;

Before looking into this function, let's recall what is a servlet. A servlet lives in a web container, and it is responsible for generating dynamic web contents. Servlet can be viewed as a special java class without main methods. After a client sends a HTTP request to the web server, the web container is responsible for :

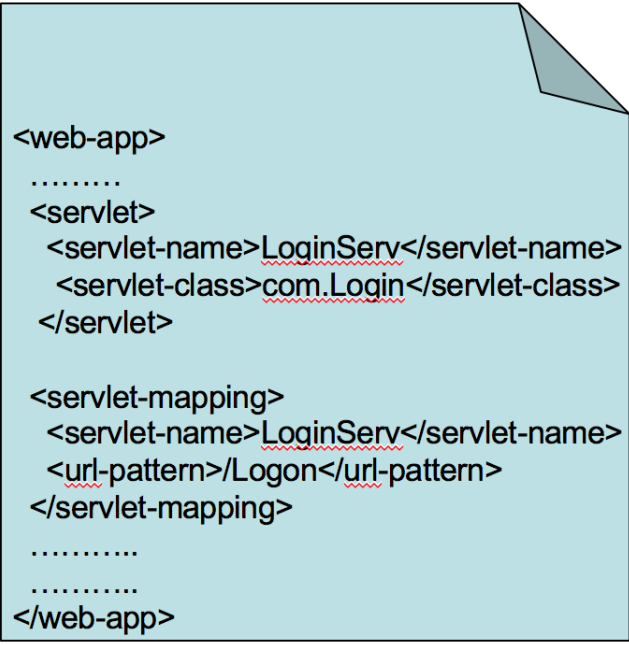
- create an instance of a servlet
- call specific method of a servlet
- destroy a servlet



The web container knows which servlet to call because a servlet can have three names :

- Client knows URL name
- Deployer knows servlet secret internal name
- Actual java class name

The XML document is responsible for deployment.



```
<web-app>
.....
<servlet>
  <servlet-name>LoginServ</servlet-name>
  <servlet-class>com.Login</servlet-class>
</servlet>

<servlet-mapping>
  <servlet-name>LoginServ</servlet-name>
  <url-pattern>/Logon</url-pattern>
</servlet-mapping>
.....
.....
</web-app>
```

Web.xml

Now let's look at what function role is this class. **ServletRequest** defines an object to provide client request information to a servlet. The servlet container creates a **ServletRequest** object and passes it as an argument to the servlet's service method. A **ServletRequest** object provides data including parameter name and values, attributes, and an input stream.

This java class file is used to form the parameter.



## **3 Check List**

### **3.1 ProductionRun.java**

#### **3.1.1 Naming Conventions**

#### **3.1.2 Indention**

#### **3.1.3 Braces**

#### **3.1.4 File Organization**

#### **3.1.5 Wrapping Lines**

#### **3.1.6 Comments**

#### **3.1.7 Java Source Files**

#### **3.1.8 Package and Import Statements**

#### **3.1.9 Class and Interface Declarations**

#### **3.1.10 Initialization and Declarations**

#### **3.1.11 Method Calls**

#### **3.1.12 Arrays**

#### **3.1.13 Object Comparison**

#### **3.1.14 Output Format**

#### **3.1.15 Computation, Comparisons, and Assignments**

#### **3.1.16 Exceptions**

#### **3.1.17 Flow of Control**

#### **3.1.18 Files**

## 3.2 ViewerServletRequest

### 3.2.1 Naming Conventions

### 3.2.2 Indention

### 3.2.3 Braces

### 3.2.4 File Organization

### 3.2.5 Wrapping Lines

### 3.2.6 Comments

### 3.2.7 Java Source Files

### 3.2.8 Package and Import Statements

### 3.2.9 Class and Interface Declarations

### 3.2.10 Initialization and Declarations

### 3.2.11 Method Calls

### 3.2.12 Arrays

### 3.2.13 Object Comparison

### 3.2.14 Output Format

### 3.2.15 Computation, Comparisons, and Assignments

### 3.2.16 Exceptions

1. The relevant exception is caught.

```
try {
    reportFileUrl = FlexibleLocation.resolveLocation(reportParam, loader);
} catch (MalformedURLException e) {
    Debug.logError(e, module);
}
if (reportFileUrl == null) {
    throw new IllegalArgumentException("Could not resolve location to URL: "
    + reportParam);
}
```

The function in this code block tries to locate the file url. In case of wrong url and no file found, an exception is raised.

### **3.2.17 Flow of Control**

No **switch** and **loop** in this file.

### **3.2.18 Files**

This java class does not deal with file operations.